electronics

MDPI

Article

SCNN-Attack: A Side-Channel Attack to Identify YouTube Videos in a VPN and Non-VPN Network Traffic

Muhammad U. S. Khan [1,*], Syed M. A. H. Bukhari [1], Tahir Maqsood [1], Muhammad A. B. Fayyaz [2], Darren Dancey [3] and Raheel Nawaz [2]

[1] Abbottabad Campus, COMSATS University Islamabad, Abbottabad 22044, Pakistan; ammar@cuiatd.edu.pk (S.M.A.H.B.); tmaqsood@cuiatd.edu.pk (T.M.)
[2] OTEHM Department, Business School, Manchester Metropolitan University, Manchester M15 6BH, UK; m.fayyaz@mmu.ac.uk (M.A.B.F.); r.nawaz@mmu.ac.uk (R.N.)
[3] Department of Computing and Mathematics, Manchester Metropolitan University, Manchester M15 6BH, UK; d.dancey@mmu.ac.uk
* Correspondence: ushahid@cuiatd.edu.pk

Abstract: Encryption Protocols e.g., HTTPS is utilized to secure the traffic between servers and clients for YouTube and other video streaming services, and to further secure the communication, VPNs are used. However, these protocols are not sufficient to hide the identity of the videos from someone who can sniff the network traffic. The present work explores the methodologies and features to identify the videos in a VPN and non-VPN network traffic. To identify such videos, a side-channel attack using a Sequential Convolution Neural Network is proposed. The results demonstrate that a sequence of bytes per second from even one-minute sniffing of network traffic is sufficient to predict the video with high accuracy. The accuracy is increased to 90% accuracy in the non-VPN, 66% accuracy in the VPN, and 77% in the mixed VPN and non-VPN traffic, for models with two-minute sniffing.

Keywords: VPN traffic classification; YouTube video identification; side-channel attack; Sequential Convolutional Neural Network—SCNN

Citation: Khan, M.U.S.; Bhukhari, S.M.A.H; Maqsood, T.; Fayyaz, M.A.B.; Dancey, D.; Nawaz, R. SCNN-Attack: A Side-Channel Attack to Identify YouTube Videos in a VPN and Non-VPN Network Traffic. *Electronics* **2022**, *11*, 350. https://doi.org/10.3390/electronics11030350

Academic Editors: Dongkyun Kim, Qinghe Du, Mehdi Sookhak, Lei Shu, Nurul I. Sarkar, Jemal Abawajy and Francisco Falcone

Received: 8 December 2021
Accepted: 21 January 2022
Published: 24 January 2022

1. Introduction

In the era of Industry 4.0 and more efficient Internet connectivity worldwide, video streams are also increasing rapidly. Many statistics indicate this growth; for example, Cisco mentions that 88% of internet traffic will be comprised of video streams by 2022 [1], Business Fortune Insights (https://www.fortunebusinessinsights.com/video-streaming-market-103057, (accessed on 1 October 2021)) predicts a growth in the global video streaming market size from USD 342.44 billion in 2019 to USD 849.93 billion in 2027. YouTube—Google's video streaming service—is among the most popular video streaming services, where, in the USA alone, more than 80% of digital video viewers use YouTube for watching online videos (https://www.emarketer.com/content/us-digital-video-2019 (accessed on 2 October 2021)). Further, a report from YouTube indicates an increase of 2.5× in online watch time (https://pk.mashable.com/tech/5230/the-livestreaming-boom-isnt-slowing-down-anytime-soon (accessed on 3 October 2021)).

With such popularity and the common access of video services by almost everyone around the world, security concerns [2,3] are inevitable. For example, fast-paced video streaming services has resulted in a fast spread of hate and misinformation in societies [4]. Various research studies and reports strongly suggest that YouTube acts as a platform to radicalize people [5]. The investigation reports from a shooting incident in a Christchurch Mosque also discusses the role of YouTube in promoting hate against the community and radicalization of the shooter (https://www.theverge.com/2020/12/8/22162779/christchurch-shooter-youtube-mosque-radicalized (accessed on 4 October 2021)). The viewing content can play an important role in identifying and profiling people,

which can help the relevant organisation to track any unusual behaviour or people with radicalized mindset or inclination.

Therefore, the need to identify the video with such content in any given network traffic is crucial. Moreover, many private enterprises, in order to save the bandwidth of their productive work, would require blocking unwanted videos on the network. For example, many private enterprises would like to block access to political, pornographic, and violence-related videos from their network. However, they would allow many other videos for their business requirements.

Traditionally, video identification systems use filters based on IP packet headers, content information, and ports to block the videos. With the popularity of content identification in network traffic by security agencies and other middlemen, video streaming service providers start encrypting the traffic due to privacy concerns. Currently, almost all the traffic between websites and clients is protected by Secure Sockets Layer (SSL)/Transport layer Security (TLS) encryption technology over the HTTPS protocol [6]. Therefore, the traditional techniques including the Deep Packet Inspection (DPI) become ineffective in identifying the individual videos in the network traffic. In addition, the use of Virtual Private Networks (VPNs) by clients and criminals makes the job of security agencies more difficult. As the traditional methods fail to identify individual videos in the network traffic, there is a need to develop new methods that can detect the videos in encrypted network traffic under VPN and non-VPN settings. This is the main problem addressed in this paper to identify the videos in encrypted network traffic and to identify them in the smallest possible time without breaking the encryption.

Therefore, the present work proposes a side-channel attack to identify videos in a VPN and non-VPN network traffic between YouTube servers and the clients. This attack verifies the assumption that, even if the network traffic is encrypted or the clients watch videos using a VPN, these videos leave their distinctive patterns, which can be used to identify those videos. A lot of information regarding a video can be obtained using flow-based features, such as the number of packet bursts, burst sizes, packet sizes, and even the number of packets being transferred between the server and the client. Machine learning models [7–9] can use these flow-based patterns to identify the videos. The present work uses a sequence of consecutive bytes per second (BPS) obtained from the traffic from a YouTube server to a client's computer and is used as a distinctive feature for identifying the video. The attack presents a novel approach, where a Sequential Convolutional Neural Network (SCNN) takes the sequence of BPS as an input to predict the title of the video. The results have shown the accuracy of 99% in detecting VPN vs. non-VPN traffic, 90% in the detection of videos in non-VPN traffic, 66% in the detection of videos in VPN traffic, and 77% accuracy when traffic traces of both VPN and non-VPN are used together for classification.

The list of contributions of this paper is as follows:

- A side-channel attack that uses a Sequential Convolutional Neural Network is proposed to identify videos in a VPN and non-VPN network traffic.
- A sequence of bytes per second (BPS) from two-minute video traffic is shown as a feature to identify the video.
- The paper demonstrates that even one-minute sniffing of network traffic can help in predicting the YouTube videos with high accuracy.

The rest of the paper is organized as follows. Section 2 presents a background of video streaming, encryption, and VPN. The related works are discussed in Section 3 and the methodology is detailed in Section 4. Experiments and results are presented in Section 5, while Section 6 concludes the paper.

2. Background

2.1. DASH Video Streaming

Since 2011, Dynamic Adaptive Streaming over HTTP (DASH) is an international standard for adaptive video streaming, where YouTube uses an advanced variant named Mov-

ing Picture Expert Group-Dynamic Adaptive Streaming over HTTP (MPEG-DASH) [10]. In adaptive video streaming, the streaming services divide a video into small segments of multiple quality (bitrates) levels that are downloaded at the client's computer in multiple requests. This methodology provides flexibility to adjust the video quality at the client side according to the network traffic condition. When a streaming session starts, the server sends information about the segments and available qualities, such as 1080p, 360p, or auto-mode quality. By default, the auto-mode quality is selected by the YouTube player at the client end. Based on the estimate of the available network throughput between the client and the server, the player requests the different quality levels of upcoming segments. Moreover, all the DASH video streaming services use Variable Bitrate (VBR) encoding that encodes videos depending on variations in the content. For example, a fast action scene is typically encoded with a high bitrate as many content changes per frame, whereas a slow-changing scene is encoded with a low bitrate as content does not change for several frames. Consequently, the DASH segments with roughly the same duration differ from each other in sizes (bytes). The consecutive DASH segments in a video and their sizes leak unique patterns for the video. Even if the video is encrypted or viewed through VPN, the videos still carry these characteristics. In this paper, the procedure to obtain the information of segments through a sequence of network traffic that traces and uses them to identify the video, is termed a side-channel attack.

2.2. Virtual Private Networks (VPNs)

Both VPN and SSL protocols hide the content of network traffic from viewing the content of the packets by a middleman between the two communicating parties. However, in an SSL only the payload of the packet is encrypted, whereas, in a VPN, the whole packet is encapsulated within another packet. The encapsulation helps to access the servers banned at the gateway by tunneling the traffic through a remote machine. A VPN does not provide security itself; rather, the VPN uses different types of security protocols to hide the content. SSL VPN and IPsec VPN are two popular types of VPNs. This work has experimented with SSL VPN provided by OpenVPN [11]. The OpenVPN uses a Hash-based Message Authentication Codes (HMAC) with a SHA1 hashing algorithm for ensuring the integrity of the contents in the packets. The encryption of the tunneled packets is ensured using a Blowfish secret-key block cipher [11].

3. Related Works

In 2011 (https://qz.com/246441/what-is-https-and-why-does-google-like-it-so-much/ (accessed on 1 October 2021)), Google introduced HTTPS as a default protocol for Gmail, which increased Youtube traffic by 97% (https://youtube-eng.googleblog.com/2016/08/youtubes-road-to-https.html (accessed on 1 October 2021) to use HTTPS encryption until 2016. Google promoted HTTPS by giving a higher ranking to websites with HTTPS. This promotion created an interest in finding methods and attacks to bypass this security protocol; therefore, a majority of research is published in this area focusing on fingerprinting the websites and applications at the clients' end to identify them in network traffic. However, several other research studies focus on identifying the videos on network traffic as well.

Initially, the research on video identification focuses on Quality of Experience (QoE) metrics to better utilize the network's bandwidth. Mangla et al. [12] predict QoE metrics of video streams using the packet headers in the network traffic, whereas Wassermann et al. [13] use a set of statistical features, such as the number and size of packets, to predict the resolution and bitrate of video streams. Statistical features are also used by Liu et al. [14] to identify traffic patterns. Gutterman et al. [15] explore the chunk statistics, such as chunk duration and chunk size, along with flow statistics, such as flow duration and direction, to predict the quality metrics for YouTube encrypted videos. Chunk statistics are also used to identify variable bitrate adaption under HTTP and QUIC protocol by Xu et al. [16].

Ameigeiras et al. [17] present a detailed analysis of YouTube streams, where the authors describe a burst characteristic feature in the YouTube videos. Their research describes the

behaviour where videos start downloading with an initial big burst but later the rate of the download decreases with small bursts. The burst characteristic feature is also discussed by Rawattu and Balasetty [18]. Dublin et al. [19] present a feature named Bits per Peak (BPP) representing traffic sizes of different bursts in a sequence. Using this feature, the authors find similarities among traffic traces of videos. Similar work is proposed in [20] with a special focus on traffic generated by a Chrome browser. The On-Off periods generated due to gaps among bursts in video streaming traffic is explored by Rao et al. [21]. Liu et al. [22] also use an On-Off mode to detect videos using machine learning algorithms. Apart from a burst size, the size of video segments is also used as a feature to create video fingerprints as discussed by Wu et al. [23]. Song et al. [24] study the role of open metadata and a storyboard linked with each YouTube video in the detection of a video in network traffic. Li et al. [25] use the video segment size, inter-request time, and download link payload size as features to detect a video in network traffic.

The above-mentioned features are used by different machine learning algorithms, such as K-nearest neighbor [13,14,20,22], SVM [20], and Neural Networks [20,22,25] as classifiers for identifying the videos with known titles. Apart from using these classification techniques, a few studies also explore the use of clustering techniques to detect the videos in network traffic where titles are unknown. Davir et al. [26] extracted the BPPs from the video traffic traces and develop an embedding/vector for each BPP similar in concept to word embeddings in Natural Language Processing. The videos are clustered based on the similarities among the embedding of difference traces. Biernacki [27] uses Min-Max, K-means, Optics, and AutoClass for clustering the videos into groups.

Identification of videos with additional encrypted traffic e.g., VPN and Tor are also researched in some studies. For example, Shi investigates the flow statistics, such as packet arrival time, intervals, and inter-packets delay for identification of videos in VPN settings [28,29]. Rahman et al. [30] evaluate the detection of videos in Tor traffic. A Tor is like a VPN; however, in a VPN, traffic is tunneled through a central server, whereas in a Tor, the traffic is tunneled through a decentralized layer of independent nodes.

The present work is a novel approach, where the model uses a sequence of bytes per second as a feature to identify the YouTube videos both in a VPN and non-VPN traffic instead of manually crafting the features. Earlier research papers identify the video by sniffing the traffic for more than 3 min, whereas the present work identifies the videos with high accuracy using only one minute of traffic sniffing.

4. Methodology

This section discusses the methodology used for the proposed technique. The threat model explains the process to deploy the attack. The attack requires the extraction of features from the raw traffic between the YouTube server and the client. The extracted features are used as input in the CNN model. The data captured for experiments and performance measures is discussed in the last subsection of methodology. The experiments and results are discussed in the next section.

4.1. Threat Model

The proposed threat model is presented in Figure 1, where the model assumes that an adversary (ISP or middleman between a client and a server) can sniff packets on a network. The adversary setups a dummy client in the network and requests the videos of interest from the server, and afterwards sniffs the communication of the client on the network. For each video of interest, the adversary requests the YouTube server multiple times one after the other and saves the network communication as pcap files along with the titles of the videos.

The adversary extracts the features from the captured pcap files and trains a machine learning model using these extracted features. The trained model is further used to predict the label (title of the video) by sniffing the network traffic of the targeted client.

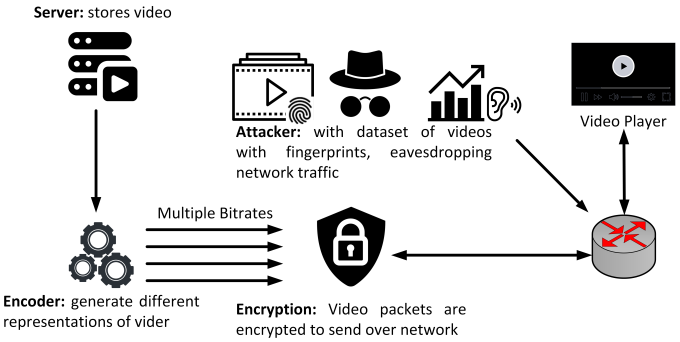

Figure 1. SCNN-Attack: Threat Model.

4.2. Feature Extraction

For classification, the set of features used is very important; hence, their selection plays a crucial role in the whole process [31]. The data in each captured pcap file consists of the initial two minutes of the video. The data in the files are grouped into uplink and downlink depending upon the source and destination IP addresses of the client. A sequence of BPS is extracted from downlinks in the files, where each sequence of BPS differs from others in every file irrespective of a VPN or non-VPN traffic due to network traffic conditions. Figure 2 shows VPN and non-VPN traffic traces (BPS) of a single video in an auto-mode quality. The traffic traces differ from each other even if the same video is requested in a fixed quality mode (Figure 3).

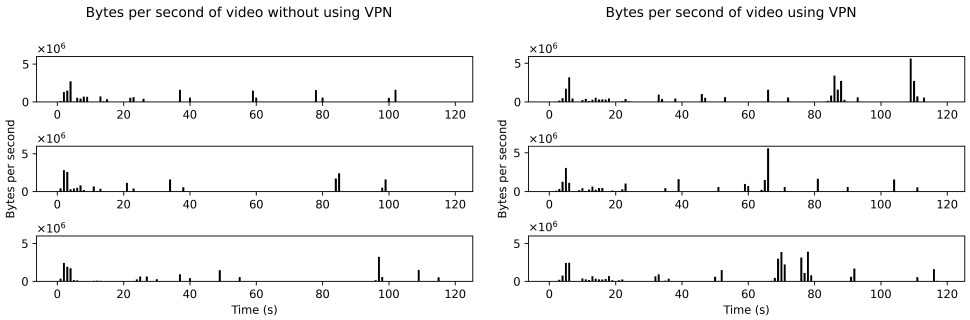

Figure 2. Six traffic traces (BPS) of the same video (https://www.youtube.com/watch?v=ZzXGSFVbVvU (accessed on 1 October 2021)) in auto-mode quality.

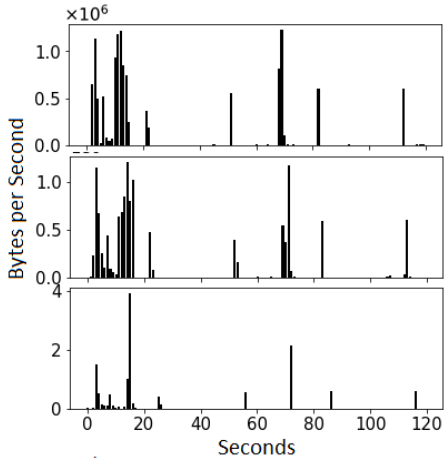

Figure 3. Three traffic traces (BPS) of the same video (https://www.youtube.com/watch?v=bDkYvUQYraU (accessed on 1 October 2021)) in 1080p quality.

This work also analyzes the variations in the sequences of BPS of a video in the whole dataset. Figure 4 represents a distribution of average BPS over consecutive 20 s in different

traces of a single video in an auto-mode quality as boxplots in VPN and non-VPN traffic. The sequences of BPS differ in each file irrespective of a VPN or non-VPN traffic due to the network traffic conditions.

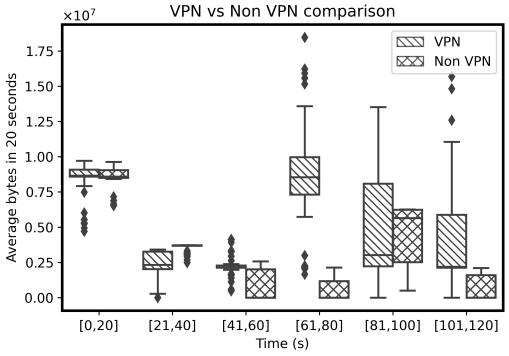

Figure 4. Average BPS over 20 s of a same video (https://www.youtube.com/watch?v=ZzXGSFVbVvU (accessed on 1 October 2021)) in auto-mode quality.

The distribution of an average BPS for different videos are compared to each other. Interestingly, the variations in the average BPS for different videos also differ from each other in the VPN (Figure 5) and non-VPN traffic (Figure 6). Due to variations within different sequences of BPS, creating further hand-crafted features is not useful on the sequence of BPS. Therefore, the sequence of BPS is selected as a feature as an input for a classifier. The sequences of BPS are normalized between zero and one before being fed to the classifier for automatic extraction of relations among different BPS sequences of the same video. The classifier uses the automatically extracted features to identify the videos.

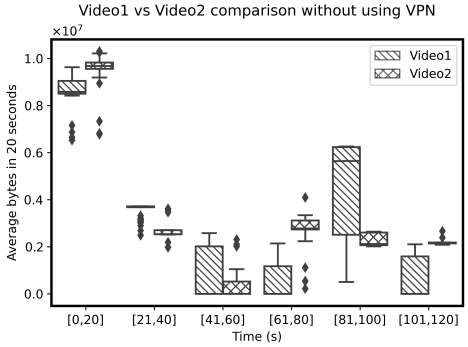

Figure 5. Average BPS over 20 s of two different videos (https://www.youtube.com/watch?v=ZzXGSFVbVvU, https://www.youtube.com/watch?v=db1u06fojyc (accessed on 1 October 2021)) in auto-mode quality.

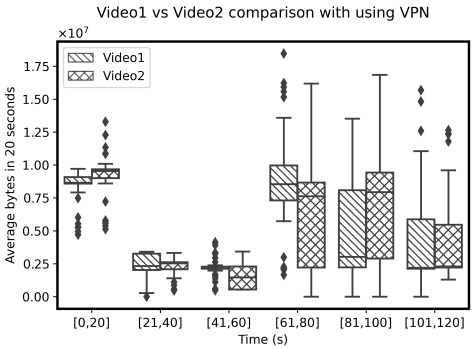

Figure 6. Average BPS over 20 s of two different videos (https://www.youtube.com/watch?v=ZzXGSFVbVvU, https://www.youtube.com/watch?v=db1u06fojyc (accessed on 1 October 2021)) in auto-mode quality with VPN.

4.3. CNN Model

The Convolutional Neural Network has been successful in many classification tasks, such as in natural language processing (hate speech detection [4,32,33], bot detection [34]), human activity recognition [35], malware detection [36], and image processing. The proposed model does not require additional hand-crafted features for the classification of videos but learns the patterns within the sequence of BPS itself. The proposed Sequential Convolutional Neural Network (SCNN) model consists of three convolutional 1D layers, three maxpooling 1D layers, dropout, flatten, and a fully connected Dense layer. The convolutional layer uses a set of independent kernels (filters), where each kernel is independently convoluted with the input data and generates a feature map as an output. The size of the kernel is set smaller than the size of the input so that the model can learn more feature maps improving the overall efficiency of the learning process.

The result of convolution passes through an activation function before being forwarded to the maxpooling layer. The pooling layer replaces the output generated by the previous layer with a summary of the nearby outputs. This replacement reduces the size of the data representation without losing the key features. Different pooling layers are being used, where the maxpooling layers take the neighbouring values from a subset of the previous input and replace it with the maximum value.

The dropout layer randomly takes the output of the previous layer as input and sets the value as zero with a pre-defined probability. This dropout technique helps prevent overfitting and not letting our model depend on certain inputs. The flatten layer in the model flattens the data for a fully connected layer—Dense layer. In a typical fully connected layer, all neurons are connected to the neurons in the previous layer. The last fully connected layer is also known as the Output Layer [37].

The input of the SCNN model is an array (1-dimensional) of BPS with a size of 120. The first convolutional layer consists of 1024 kernels of size 6 with a stride of 1. Each neuron in the layer is connected to a neighbourhood of 6 elements in the input data. The outputs of the layer are 1024 feature maps of size 120. The first convolutional layer contains a total of 7168 trainable parameters (6144 weights and 1024 bias parameters). The next layer is a maxpooling layer that comprises 1024 feature maps of size 60. Each feature map of this maxpooling layer is connected to two feature maps of the previous convolutional layer. The maxpooling layer requires zero trainable parameters.

The second convolutional layer with 512 kernels of size 4 contains a total number of 2,097,664 trainable parameters gives an output of 512 feature maps of size 60. The second convolutional layer is followed by a second maxpooling layer with 512 feature maps of size 30. The third convolutional layer with 1,311,232 trainable parameters consists of 512 kernels of size 5. The outputs of the third convolutional layer are 512 feature maps of size 30. Consequently, the third maxpooling layer generates 512 feature maps of size 15. After the third maxpooling layer, the proposed model has a dropout layer that drops the values of neurons from the previous layer with a probability of 0.2. The flatten layer comes after the dropout layer and it converts the 512 feature maps from the previous layer to a one-dimensional feature of size 7680. The last fully connected layer comprises 337,964 trainable parameters in 44 different classes that are used in the data.

All the convolutional layers in the proposed model use a ReLU activation function, whereas the last fully connected layer uses a softmax activation function. The ReLU activation function outputs the input values if input values are positive; otherwise, the function provides zero as an output. The softmax activation function is a generalization of logistic regression. The softmax function normalizes a D-dimensional vector of real values to a D-dimensional probability distribution vector of real values in the range [0, 1] that sums up to 1. The D represents the number of output classes and each value in the D-dimensional output vector presents a probability score of the corresponding class.

The cost function, categorical cross-entropy, is used in the proposed model. The cost function measures the difference between the softmax layer's output and the true label (in a hot-encoding format) of a sample being trained. Moreover, for an optimization task, the

model uses an Adam optimization algorithm [38]. The architecture of the SCNN model is presented in Figure 7. Table 1 presents the summary of the model.

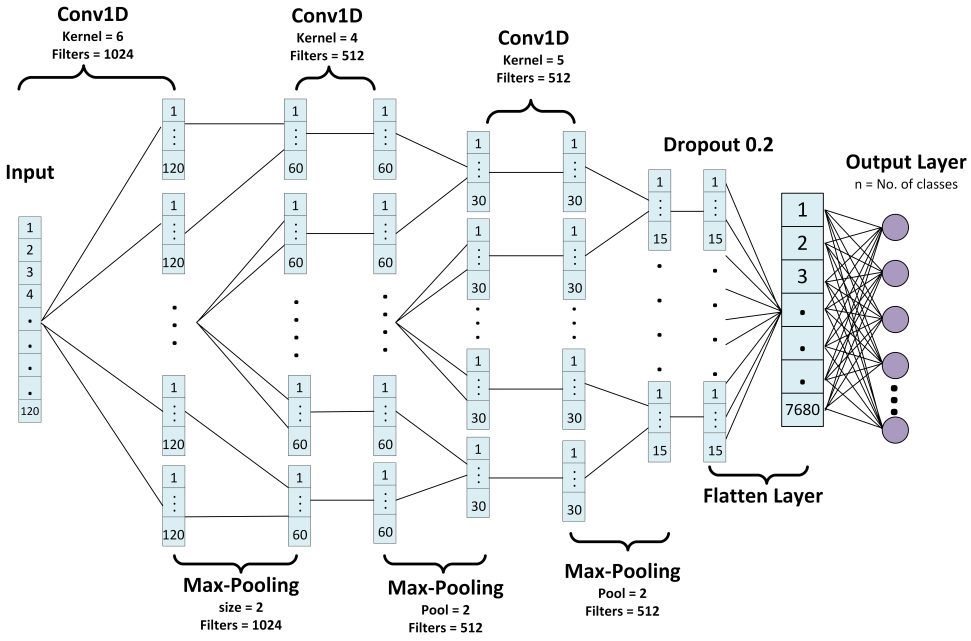

Figure 7. SCNN Model: Architecture.

Table 1. SCNN Model Summary for 360p quality videos dataset.

Layer ID	Layer (Type)	Output Shape	Param #
1	Conv1D	(None, 120, 1024)	7168
2	MaxPooling1	(None, 60, 1024)	0
3	Conv1D	(None, 60, 512)	2,097,664
4	MaxPooling1	(None, 30, 512)	0
5	Conv1D	(None, 30, 512)	1,311,232
6	MaxPooling1	(None, 15, 512)	0
7	Dropout	(None, 15, 512)	0
8	Flatten	(None, 7680)	0
9	Dense	(None, Number of videos = 44)	337,964

Total params: 3,754,028; Trainable params: 3,754,028; Non-trainable params: 0.

4.4. Dataset and Performance Evaluation

OpenVPN with an external VPN desktop client—SurfShark—is used as the dummy client for VPN traffic [39]. The Selenium web tool along with a chrome driver is used to automate the dummy client on the network to request the video of interest from the YouTube server. The network traffic is captured using Python's TShark library and each video communication is stored as a pcap file along with the video title. The client requests the YouTube videos in three different modes: 1080p, 360p, and auto-mode. YouTube presents videos of the highest quality in 1080p and lowest quality in 360p. The auto-mode of YouTube varies the quality of the videos depending upon the bandwidth availability at the client-side. The collected dataset consists of 43 different videos for auto-mode, 44 for 360p, and 20 for 1080p quality modes. Each video is requested 50 times using a VPN and 50 times without a VPN to create multiple instances. The initial two minutes of the videos are captured and converted into pcap files for experimentation purpose.

However, for detecting the video from anywhere on its timeline, the attacker needs to store the whole video.

For each quality mode, the instances are divided randomly into training and testing datasets with a split ratio of 60:40. The performance of the algorithm is evaluated using the accuracy measure and confusion matrix. The accuracy measure evaluates the performance of the algorithm over all the classes, whereas the confusion matrix is used to evaluate the performance over each class. The accuracy measure is defined as follows:

$$Accuracy = \frac{Correct\ Predictions}{Total\ Predictions} \quad (1)$$

5. Experiments and Results

Six different scenarios are set up for an evaluation of the proposed SCNN technique, where Scenario A, Scenario B, and Scenario C execute the technique on 1080p, 360p, and auto-mode quality datasets. In Scenario D, the traffic traces of videos in auto-mode quality are tested in VPN traffic. All the traces of videos in auto-mode quality in the VPN traffic are labelled under one class (VPN) instead of their labels, and the traces of videos in auto-mode quality in the non-VPN traffic are labelled as non-VPN in Scenario E. However, in Scenario F, the traffic traces are assigned their original video labels and identified in a mixed VPN and non-VPN traffic.

The proposed SCNN technique is compared with the following machine learning algorithms: (1) Sequential minimal optimization and (2) Naïve Bayes. Sequential minimal optimization (SMO) is a popular machine learning algorithm to solve quadratic programming (QP) problems and train the Support Vector Machines (SVM). The algorithm divides the problem into a series of small parts and solves the problem analytically. Naïve Bayes algorithm is a probabilistic model of classification that depends upon the Bayesian theorem. The algorithm assumes that each feature in the input is independent and does not depend upon any other feature in the data.

The algorithms are compared for different sizes of traffic traces in all the scenarios. Initially, the models are trained and tested for a size of 120 s. However, 20, 40, 60, 80, and 100 s of traffic traces are used to find the smallest suitable size for detecting the video titles in the traffic traces.

Figure 8 presents the accuracy of the proposed model in Scenario A. The confusion matrix of the experiment is shown in Figure 9. The proposed SCNN model achieves the testing accuracy of 60% on a 120 s traffic trace better than the Naïve Bayes and SMO algorithms. However, the accuracy drops below 40% on the traffic traces of the size 40 s or less. The Naïve Bayes and SMO also presented a similar behaviour. The comparison results of the SCNN, Naïve Bayes, and SMO are presented in Figure 10.

In Scenario B, the models are evaluated on a 360p quality videos dataset. The 360p dataset is a larger dataset than the 1080p quality videos dataset in terms of the number of videos (labels) and the number of trace files. The increase in dataset results in an increase in the testing accuracy of the model (Figure 11). More classes (labels) are predicted accurately among all the test samples of those classes. This effect is shown as red dots in the diagonal of the confusion matrix (Figure 12). The accuracy comparison of SCNN, SMO, and Naïve Bayesian is presented in Figure 13.

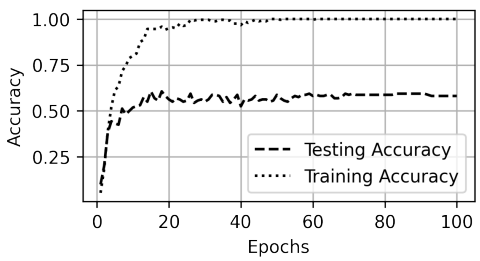

Figure 8. Accuracy on 1080p dataset.

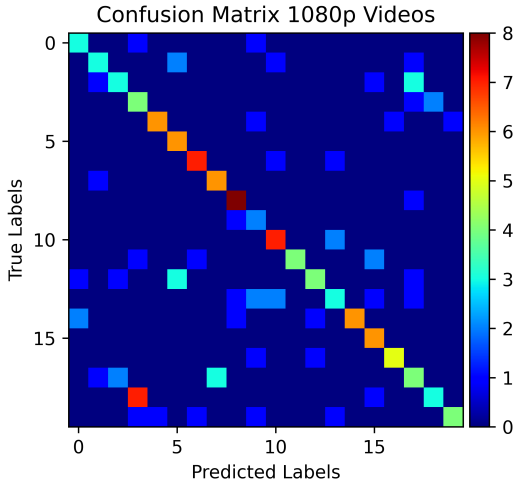

Figure 9. Confusion Matrix for 1080p dataset.

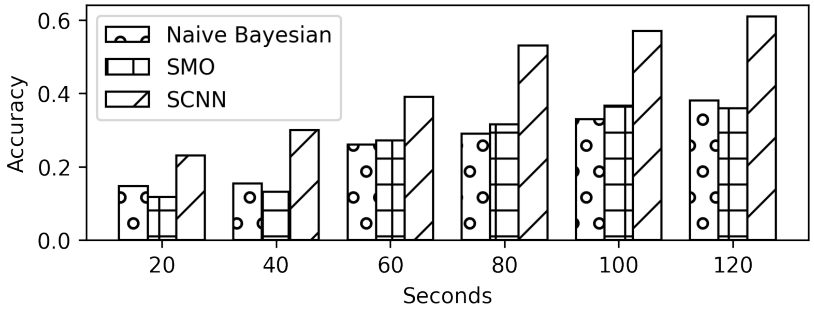

Figure 10. Comparison results against trace size (in seconds) for 1080p quality.

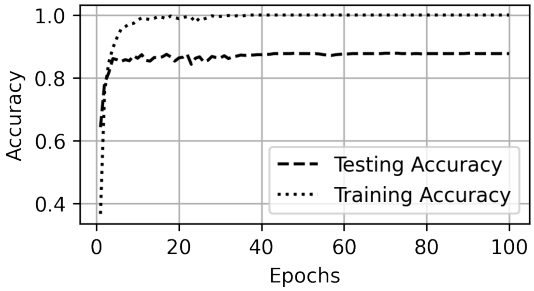

Figure 11. Accuracy on 360p dataset.

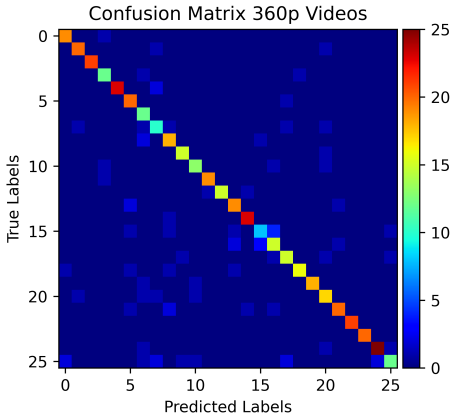

Figure 12. Confusion Matrix for 360p dataset.

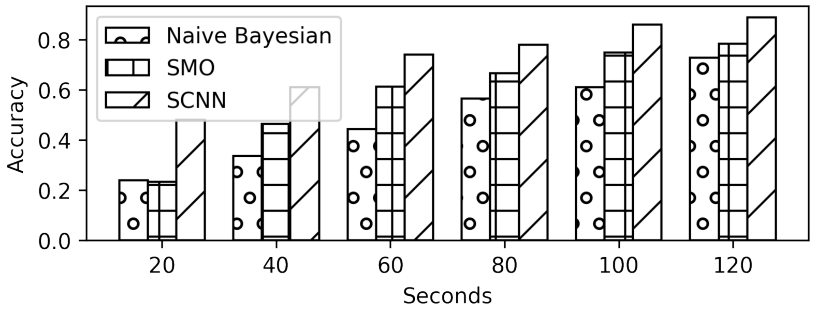

Figure 13. Comparison results against trace size (in seconds) for 360p quality.

Scenario A and Scenario B evaluate the proposed model on the fixed quality video datasets. The default method of the YouTube player is to play the videos in auto-mode quality. Scenario C evaluates the models for detection of videos in auto-mode quality. The proposed model achieves 90% accuracy (Figure 14) and more classes are predicted correctly for their tested samples as shown in Figure 15. The model performs better than both the Naïve Bayes and SMO (Figure 16).

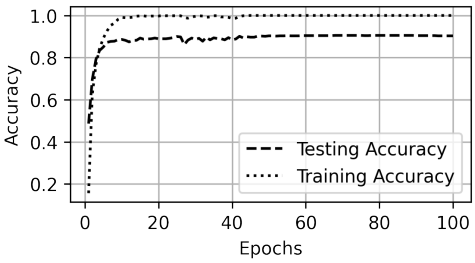

Figure 14. Accuracy on auto-mode quality.

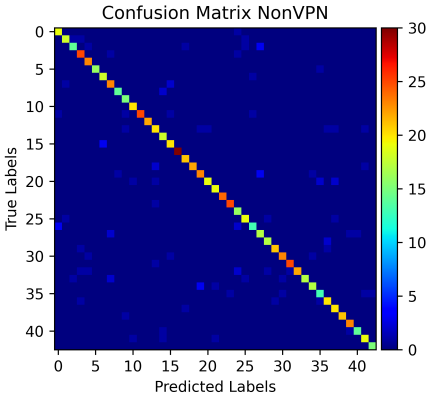

Figure 15. Confusion Matrix for auto-mode quality.

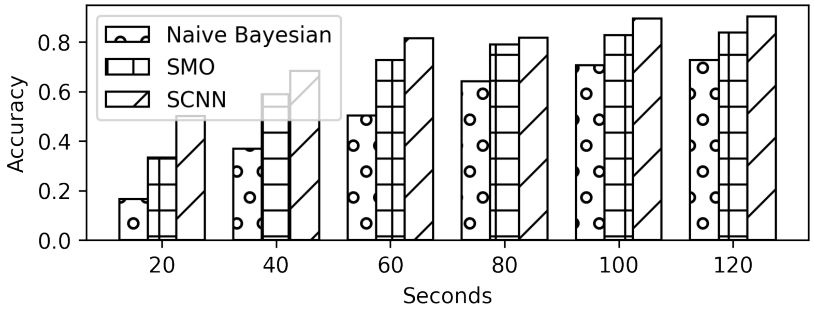

Figure 16. Comparison results against trace size (in seconds) for auto-mode quality.

The videos in the auto-mode quality are also requested while running the VPN on the system. Scenario D evaluates the performance of the models on auto-mode quality in the VPN settings. The performance of the model drops from a 90% in the non-VPN setting to a 66% testing accuracy in the VPN settings as shown in Figures 17 and 18. However, the proposed SCNN model performs better than the Naïve Bayes and SMO (Figure 19). Even with a trace size of 40 s, the proposed SCNN achieves a 50% accuracy.

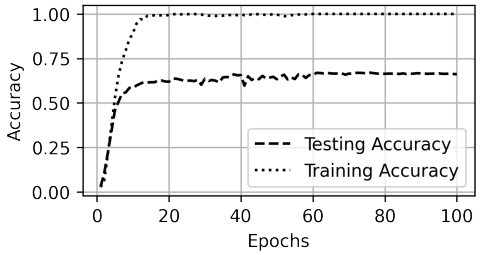

Figure 17. Accuracy on auto-mode quality in VPN dataset.

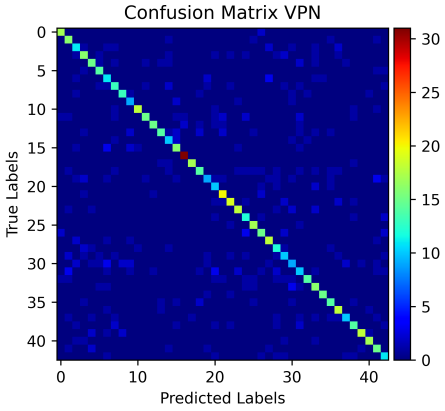

Figure 18. Confusion Matrix for auto-mode quality in VPN traffic.

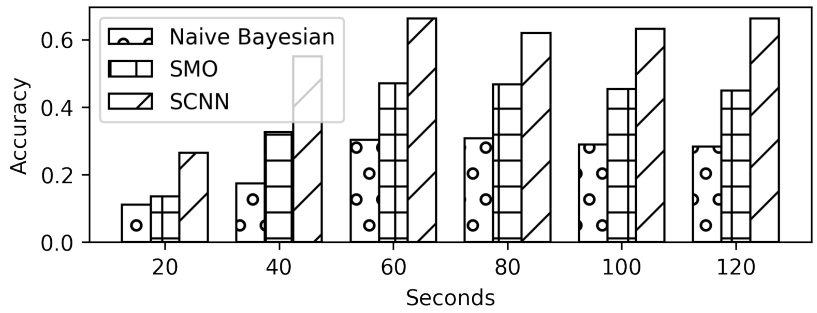

Figure 19. Comparison results against trace size (in seconds) for auto-mode quality on VPN.

In Scenario E, the models are evaluated for detection of a VPN vs. non-VPN traffic. All the traffic traces for auto-mode quality videos are assigned a single label (non-VPN). Similarly, all the traces in a VPN set are labelled as VPN instead of their class labels. The results show the high accuracy of 99% correctly identifying both the classes (Figures 20 and 21). Even the Naïve Bayes and SMO show above 95% accuracy. Only a 20-second traffic trace is enough to detect that the traffic is with or without a VPN setting with above 85% accuracy as shown in Figure 22.

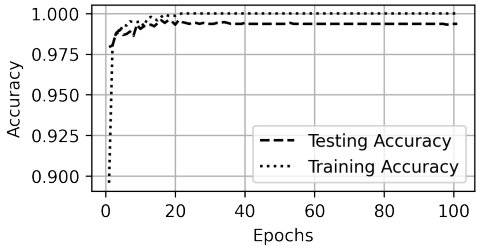

Figure 20. Accuracy for VPN and non-VPN traffic.

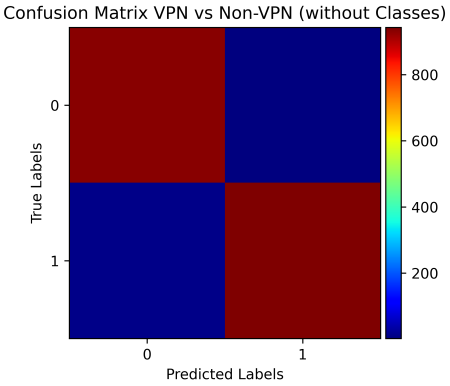

Figure 21. Confusion Matrix for VPN and non-VPN traffic.

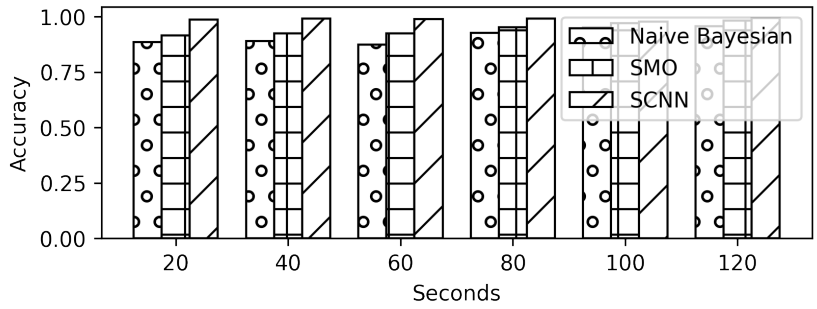

Figure 22. Comparison results for VPN and non-VPN traffic.

Scenario F presents a case when the attacker tries to detect the video without the information that traffic is a VPN or non-VPN. In this scenario, the original labels are assigned to the traffic traces with the modification that along with the label, information regarding a VPN setting is added to the label. Therefore, if a Label_1 is in a non-VPN setting, the label becomes Label_1_non-VPN and if it is in VPN settings, it becomes Label_1_VPN. The results show a decrease in the testing accuracy to 77% (Figures 23 and 24) from 99% in Scenario E and 90% from only non-VPN settings in the Scenario C. However, the accuracy remains better than the 60% accuracy of Scenario D. In Scenario F, only 40 s traffic traces prove to be sufficient to predict the label with above 60% accuracy using the proposed methodology as shown in Figure 25.

In all the scenarios, the proposed model performs better than Naïve Bayes and SMO. The results have shown that YouTube videos can be detected by the adversary who can sniff the network traffic in both the VPN and non-VPN traffic.

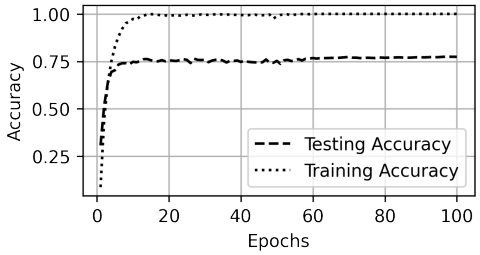

Figure 23. Accuracy on all auto-mode quality videos in VPN and non-VPN datasets.

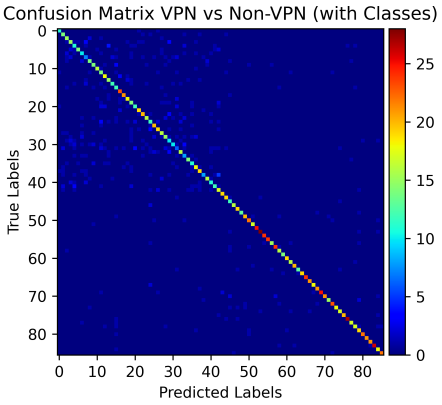

Figure 24. Confusion Matrix on all auto-mode quality videos in VPN and non-VPN datasets.

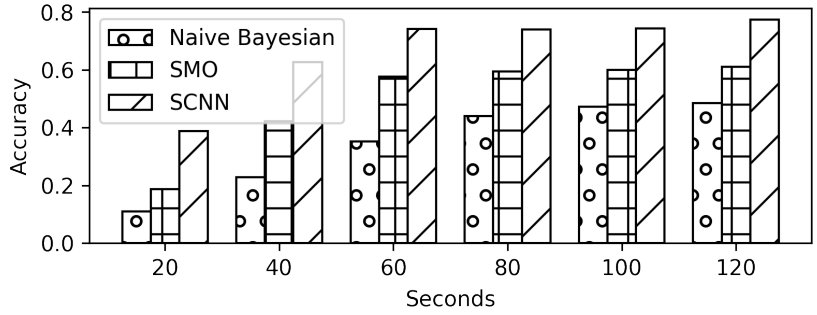

Figure 25. Accuracy results for all videos in VPN and non-VPN traffic.

6. Conclusions and Future Works

This study presents a side-channel attack named SCNN-Attack. The SCNN-Attack utilizes a Sequential Neural Network model to predict the video stream in a VPN and non-VPN network traffic. A sequence of bytes per second from a two-minute sniffing of network traffic can identify a video with high accuracy. The results have shown that HTTPS and VPN are not enough to hide the identity of the videos in the network traffic. Any adversary, such as a network administrator or ISP can identify the YouTube videos being watched by the clients on the network.

The proposed attack is limited due to the problem of concept drift that is caused due to the instability of the network. The attack requires active learning with retraining after every week. Therefore, the proposed technique requires huge computational and space requirements at the adversary's end that keeps the attack limited for large scale deployment. In future works, concept drift handling and detection techniques along with the attack will increase the practical deployments of the attack.

Author Contributions: Methodology, M.U.S.K., T.M. and M.A.B.F.; Formal Analysis, M.U.S.K. and S.M.A.H.B.; Project Administration, M.U.S.K.; Data Curation, S.M.A.H.B.; Writing—Review and Editing, M.A.B.F.; Supervision, D.D. and R.N. All authors have read and agreed to the published version of the manuscript

Funding: This research received no external funding.

Acknowledgments: This material is based upon a work supported by the National Centre of Cyber Security (NCCS), Pakistan and Higher Education Commission (HEC) under grant RF-NCCS-023. The authors also thank Shouzeb Hassan, Waleed Afandi, and Ali Sher from the Department of Computer Science, COMSATS Abbottabad, for correcting this document.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Gu, J.; Wang, J.; Yu, Z.; Shen, K. Walls have ears: Traffic-based side-channel attack in video streaming. In Proceedings of the IEEE INFOCOM 2018-IEEE Conference on Computer Communications, Honolulu, HI, USA, 16–19 April 2018; pp. 1538–1546.
2. Ghayvat, H.; Pandya, S.N.; Bhattacharya, P.; Zuhair, M.; Rashid, M.; Hakak, S.; Dev, K. CP-BDHCA: Blockchain-based Confidentiality-Privacy preserving Big Data scheme for healthcare clouds and applications. *IEEE J. Biomed. Health Inform.* **2021**. [CrossRef] [PubMed]
3. Mishra, N.; Pandya, S. Internet of Things Applications, Security Challenges, Attacks, Intrusion Detection, and Future Visions: A Systematic Review. *IEEE Access* **2021**, *9*, 59353–59377.. [CrossRef]
4. Khan, M.U.S.; Abbas, A.; Rehman, A.; Nawaz, R. HateClassify: A Service Framework for Hate Speech Identification on Social Media. *IEEE Internet Comput.* **2020**, *25*, 40–49. [CrossRef]
5. Ledwich, M.; Zaitsev, A. Algorithmic extremism: Examining YouTube's rabbit hole of radicalization. *arXiv* **2019**, arXiv:1912.11211.
6. Khan, M.U.S.; Bukhari, S.M.A.H.; Khan, S.A.; Maqsood, T. ISP can identify YouTube videos that you just watched. In Proceedings of the 18th International Conference on Frontiers of Information Technology (FIT 2021), Islamabad, Pakistan, 13–14 December 2021.
7. Khan, M.U.S.; Jawad, M.; Khan, S.U. Adadb: Adaptive Diff-Batch Optimization Technique for Gradient Descent. *IEEE Access* **2021**, *9*, 99581–99588. [CrossRef]
8. Irfan, R.; Khalid, O.; Khan, M.U.S.; Rehman, F.; Khan, A.U.R.; Nawaz, R. SocialRec: A Context-Aware Recommendation Framework With Explicit Sentiment Analysis. *IEEE Access* **2019**, *7*, 116295–116308. [CrossRef]
9. Zaidi, K.S.; Hina, S.; Jawad, M.; Khan, A.N.; Khan, M.U.S.; Pervaiz, H.B.; Nawaz, R. Beyond the Horizon, Backhaul Connectivity for Offshore IoT Devices. *Energies* **2021**, *14*, 6918. [CrossRef]
10. Dubin, R.; Dvir, A.; Pele, O.; Hadar, O. I know what you saw last minute—encrypted http adaptive video streaming title classification. *IEEE Trans. Inf. Forensics Secur.* **2017**, *12*, 3039–3049. [CrossRef]
11. Miller, S.; Curran, K.; Lunney, T. Detection of Virtual Private Network Traffic Using Machine Learning. *Int. J. Wirel. Netw. Broadband Technol. (IJWNBT)* **2020**, *9*, 60–80. [CrossRef]
12. Mangla, T.; Halepovic, E.; Ammar, M.; Zegura, E. Using session modeling to estimate HTTP-based video QoE metrics from encrypted network traffic. *IEEE Trans. Netw. Serv. Manag.* **2019**, *16*, 1086–1099. [CrossRef]
13. Wassermann, S.; Seufert, M.; Casas, P.; Gang, L.; Li, K. Let me decrypt your beauty: Real-time prediction of video resolution and bitrate for encrypted video streaming. In Proceedings of the 2019 Network Traffic Measurement and Analysis Conference (TMA), Paris, France, 19–21 June 2019; pp. 199–200.
14. Liu, Y.; Li, S.; Zhang, C.; Zheng, C.; Sun, Y.; Liu, Q. Itp-knn: Encrypted video flow identification based on the intermittent traffic pattern of video and k-nearest neighbors classification. In *International Conference on Computational Science*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 279–293.
15. Gutterman, C.; Guo, K.; Arora, S.; Wang, X.; Wu, L.; Katz-Bassett, E.; Zussman, G. Requet: Real-time QoE detection for encrypted YouTube traffic. In Proceedings of the 10th ACM Multimedia Systems Conference, Amherst, MA, USA, 18–21 June 2019; pp. 48–59.
16. Xu, S.; Sen, S.; Mao, Z.M. CSI: Inferring mobile ABR video adaptation behavior under HTTPS and QUIC. In Proceedings of the Fifteenth European Conference on Computer Systems, Heraklion, Greece, 27–30 April 2020; pp. 1–16.
17. Ameigeiras, P.; Ramos-Munoz, J.J.; Navarro-Ortiz, J.; Lopez-Soler, J.M. Analysis and modelling of YouTube traffic. *Trans. Emerg. Telecommun. Technol.* **2012**, *23*, 360–377. [CrossRef]
18. Ravattu, R.; Balasetty, P. Characterization of YouTube Video Streaming Traffic. 2013. Available online:Available online: https://www.diva-portal.org/smash/get/diva2:830691/FULLTEXT01.pdf (accessed on 2 December 2021).
19. Miller, B.; Huang, L.; Joseph, A.D.; Tygar, J.D. I know why you went to the clinic: Risks and realization of https traffic analysis. In *International Symposium on Privacy Enhancing Technologies Symposium*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 143–163.
20. Dubin, R.; Dvir, A.; Hadar, O.; Pele, O. I Know What You Saw Last Minute-the Chrome Browser Case. *Black Hat Europe*. 2016. Available online: https://paper.bobylive.com/Meeting_Papers/BlackHat/Europe-2016/eu-16-Dubin-I-Know-What-You-Saw-Last-Minute-The-Chrome-Browser-Case-WP.pdf (accessed on 6 December 2021).
21. Rao, A.; Legout, A.; Lim, Y.s.; Towsley, D.; Barakat, C.; Dabbous, W. Network characteristics of video streaming traffic. In Proceedings of the Seventh COnference on emerging Networking EXperiments and Technologies, Tokyo, Japan, 6–9 December 2011; pp. 1–12.

22. Liu, Y.; Li, S.; Zhang, C.; Zheng, C.; Sun, Y.; Liu, Q. DOOM: A Training-Free, Real-Time Video Flow Identification Method for Encrypted Traffic. In Proceedings of the 2020 27th International Conference on Telecommunications (ICT), Bali, Indonesia, 5–7 October 2020; pp. 1–5.
23. Wu, H.; Yu, Z.; Cheng, G.; Guo, S. Identification of Encrypted Video Streaming Based on Differential Fingerprints. In Proceedings of the IEEE INFOCOM 2020-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Toronto, ON, Canada, 6–9 July 2020; pp. 74–79.
24. Song, J.; Lee, S.; Kim, B.; Seol, S.; Lee, B.; Kim, M. VTIM: Video Title Identification Using Open Metadata. *IEEE Access* **2020**, *8*, 113567–113584. [CrossRef]
25. Li, F.; Chung, J.W.; Claypool, M., Silhouette: Identifying YouTube Video Flows from Encrypted Traffic. In Proceedings of the 28th ACM SIGMM Workshop on Network and Operating Systems Support for Digital Audio and Video, Amsterdam, The Netherlands, 12–15 June 2018; Association for Computing Machinery: New York, NY, USA, 2018; pp. 19–24.
26. Dvir, A.; Marnerides, A.K.; Dubin, R.; Golan, N. Clustering the Unknown-The Youtube Case. In Proceedings of the 2019 International Conference on Computing, Networking and Communications (ICNC), Honolulu, HI, USA, 18–21 February 2019; pp. 402–407.
27. Biernacki, A. Identification of adaptive video streams based on traffic correlation. *Multimed. Tools Appl.* **2019**, *78*, 18271–18291. [CrossRef]
28. Shi, Y. *Towards Machine Learning Based Source Identification of Encrypted Video Traffic*; Michigan State University: East Lansing, MI, USA, 2019.
29. Shi, Y.; Biswas, S. A deep-learning enabled traffic analysis engine for video source identification. In Proceedings of the 2019 11th International Conference on Communication Systems & Networks (COMSNETS), Bengaluru, India, 7–11 January 2019; pp. 15–21.
30. Rahman, M.S.; Matthews, N.; Wright, M. Poster: Video fingerprinting in tor. In Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, London, UK, 11–15 November 2019; pp. 2629–2631.
31. Khalil, H.; Khan, M.U.; Ali, M. Feature Selection for Unsupervised Bot Detection. In Proceedings of the 2020 3rd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET), Sukkur, Pakistan, 29–30 January 2020; pp. 1–7.
32. Mahmood, Z.; Safder, I.; Nawab, R.M.A.; Bukhari, F.; Nawaz, R.; Alfakeeh, A.S.; Aljohani, N.R.; Hassan, S.U. Deep sentiments in roman urdu text using recurrent convolutional neural network model. *Inf. Process. Manag.* **2020**, *57*, 102233. [CrossRef]
33. Safder, I.; Hassan, S.U.; Visvizi, A.; Noraset, T.; Nawaz, R.; Tuarob, S. Deep learning-based extraction of algorithmic metadata in full-text scholarly documents. *Inf. Process. Manag.* **2020**, *57*, 102269. [CrossRef]
34. Mohammad, S.; Khan, M.U.S.; Ali, M.; Liu, L.; Shardlow, M.; Nawaz, R. Bot detection using a single post on social media. In Proceedings of the 2019 Third World Conference on Smart Trends in Systems Security and Sustainablity (WorldS4), London, UK, 30–31 July 2019; pp. 215–220.
35. Khan, M.U.; Abbas, A.; Ali, M.; Jawad, M.; Khan, S.U. Convolutional neural networks as means to identify apposite sensor combination for human activity recognition. In Proceedings of the 2018 IEEE/ACM International Conference on Connected Health: Applications, Systems and Engineering Technologies (CHASE), Washington, DC, USA, 26–28 September 2018; pp. 45–50.
36. Khan, M.; Baig, D.; Khan, U.S.; Karim, A. Malware Classification Framework using Convolutional Neural Network. In Proceedings of the 2020 International Conference on Cyber Warfare and Security (ICCWS), Islamabad, Pakistan, 20–21 October 2020; pp. 1–7.
37. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
38. Khan, W.; Ali, S.; Muhammad, U.K.; Jawad, M.; Ali, M.; Nawaz, R. AdaDiffGrad: An Adaptive Batch Size Implementation Technique for DiffGrad Optimization Method. In Proceedings of the 2020 14th International Conference on Innovations in Information Technology (IIT), Al Ain, United Arab Emirates, 17–18 November 2020; pp. 209–214.
39. Chen, J.; Wu, J.; Liang, H.; Mumtaz, S.; Li, J.; Konstantin, K.; Bashir, A.K.; Nawaz, R. Collaborative trust blockchain based unbiased control transfer mechanism for industrial automation. *IEEE Trans. Ind. Appl.* **2019**, *56*, 4478–4488. [CrossRef]